Chapter 1

Distributed Resource Exploitation for Autonomous Mobile Sensor Agents in Dynamic Environments

Sarjoun Doumit and Ali Minai
Complex Adaptive Systems Laboratory (C.A.S.L.)
University of Cincinnati, Ohio, U.S.A.
sdoumit@ececs.uc.edu, aminai@ececs.uc.edu

This paper studies the *distributed resource exploitation problem (DREP)* where many *resources* are distributed across an unknown environment, and several agents move around in it with the goal to exploit/visit the resources. A resource may be anything that can be harvested/sensed/acted upon by an agent when the agent visits that resource's physical location. A sensory agent (SA) is a mobile and autonomous *sensory entity* that has the capability of sensing a resource's attribute and therefore determining the *exploitatory gain factor* or *profitability* when this resource is visited. This type of problem can be seen as a combination of two well-known problems: the Dynamic Traveling Salesman Problem (DTSP) [5] and the Vehicle Routing Problem (VRP) [1]. But the DREP differs significantly from these two. In the DTSP we have a single agent that needs to visit many fixed cities that have costs associated to their pairwise links, so it is an optimization of paths on a static graph with time-varying costs. In VRP on the other hand, we have a number of vehicles with uniform capacity, a common depot, and several stationary customers scattered around an environment, so the goal isto find the set of routes with overall minimum route cost to service all the customers. In our problem, we have multiple SAs deployed in an unknown environment

with multiple dynamic resources each with a dynamically varying value. The goal of the SAs is to adapt their paths *collaboratively* to the dynamics of the resources in order to maximize the general *profitability* of the system. Applications of this model range from exploratory missions such as those of rovers on planets [4, 2, 3] to surveillance, monitoring, resupply and survival in hazardous environments.

1.1 Introduction

The special case of the distributed resource exploitation problem (DREP) we consider can be seen as a scenario where a group of agents, deployed in an (unexplored) area to fulfill a certain task (including exploring the area), partition the discovered targets or"resources" into different classes in order to come up with the best path planning scheme for their application. In the existing related literature, this challenge falls into the category of *multiobjective optimization*, i.e. the many parameters the agents have to consider in order to design the optimal paths that will allow them to exploit the largest amount of resources while considering the constraints of every problem type. The constraints we consider include time, energy of the agent, communication and cost of mobility, but others could be considered. In brief, the task or problem becomes finding the vector of *decision variables* $\overrightarrow{X}_{var}$ that satisfies all the system constraints (agents, resources and environment) and still *optimizes* a vector function $\overrightarrow{V}_F$ which represents the objective function F. The functions that are members of $\overrightarrow{V}_F$ are usually conflicting due to opposing goals, such as minimize energy expenditures (in order to increase longevity of the agent) and keep visiting resources (which costs energy and decreases longevity). An important tradeoff is *exploration vs. exploitation*: should the agent go to explore unexplored parts of the terrain in the (possibly vain) hope of discovering new resources or visit already discovered resources? In the next section we discuss the system description and related constraints in more detail.

1.2 System Description

The system consists of the SAs and an "unexplored" terrain which includes the hotSpots. The SA is a relatively small sensing unit equipped with a battery for energy and has mobility and wireless communication capabilities. The SA's mobility resembles the locomotion of a fluid particle in a field of potentials (where the potentials are a result of hotSpots and unexplored terrain areas). The agent's "motor system" generates a constant moving force that is guided by a "directional steering" capable of heading in any direction ($0° \rightarrow 360°$) in 3D space. The agents incur a mobility and communication energy cost. Since the terrain we're addressing is a 3D non-homogeneous environment, the mobility cost differs when traversing different types of terrain, or going on an incline or a combination of these elements. Each SA records the energy cost of going between specific points on the terrain, so it can create its own virtual asymmetric

weighted-edge graph which is it's view of the terrain. The SA earns different "rewards" for every time it "discovers new areas", "discovers a new hotSpot" or "visits a previously discovered hotSpot". Every agents is equipped with an *algorithmic strategy* module that allows to plan its next step and even much further in the future based on the constant information it collects itself from the terrain (or from other SAs). The decision mechanism of the SA, which we will not describe in detail, is based on a subsumption architecture. Every agent has a 2-stage path-planning module with two functions:

- *Long-term planning*: draws strategies and list of future actions to be taken by the agent and the possibility/probability of certain events that could occur. Decides the value of the short-term planning's time-steps (or *Horizon* value) and informs the short-term planning of the current destination point to go to.

- *Short-term planning*: uses its specific *Horizon* time value to calculate the specific steps needed to go to a specified target and regularly updates the bearings and angle of the agent to the compensate for any path irregularities that might arise due to obstacles and different terrain textures. Note that information updates from neighboring agents cannot prompt it to alter its course until it has finished.

The potential fields represent the discovered hotSpots' status and the unexplored areas of the field or the areas that are explored but weren't visited in some time. The hotSpots are dynamic in nature and decay away after some time, each at a different rate.

1.2.1 Types of Solution Approaches

We classify the approach for distributed management and control of the agents into three classes: aware agents, semi-aware agents and unaware agents. Each of these classes defines how informed a single agent is about the rest of the agents in the network. We describe these classes below, and also show the results of the same experiments run with these three classes of agents.

- Aware agents In the aware case, every agent knows the state of every other agent on the field. Every single agent can learn and take advantage of all the decisions, discoveries and feedback by all other agents on the field. This would represent the perfect scenario as long as the cost of information (whether acquired by wireless or some other medium) is negligible. The decisions taken by these agents reflect the behavior of the entire system and follow the embedded algorithm as a monolithic system. It is, therefore, very predictable.

- Semi-aware agents In the semi-aware case, the actual distance separating the nodes and limitations on information transfer are taken into consideration. Therefore, only agents within a physical proximity of each other can

communicate and only part of their knowledge is transmitted and shared. The cost of transmission is factored in when determining the efficiency of the total system. Due to the limited information sharing, the collective behavior of the agents is not very predictable and could result in surprising solutions and many local semi-optimal solutions.

- Unaware agents The last class of agents represents true classic complex systems, similar to very primitive cellular organisms, where every agent interacts with its immediate environment and chooses its actions independently of other agents in the network. This class embodies true emergence where the system is completely unpredictable and no optimal solution is guaranteed.

All the known hotSpots exert specific "gravitational" and "anti-gravitational" forces on the agents in their vicinity. When an agent discovers a hotSpot, it assigns it a specific "profitability" value $p(t)$ which corresponds to the intensity with which its sensed phenomenon was recorded. A unique timer function $f(t)$ is also associated to every discovered hotSpot to determine the value of the "attractiveness/repulsion" of the hotSpot which is proportional to the time value and other variables. Thus, an SA that has recently been "attracted" to an unvisited hotSpot would immediately afterwards be "repulsed" by that hotSpot and, based upon the different rewards it has accumulated and on its specific exploratio/exploitation probability ratio, it would either gravitate towards an unexplored section in the terrain, or towards another hotSpot. When exchange of information occurs amongst the SAs, the network's values for every hotSpot's attractiveness, etc. becomes the deciding factor for assigning SAs to targets. The proximity of the SA to other hotSpot's and unexplored parts of the terrain, in addition to the cost of the virtual edges, play the role in an "optimized preferential matching" algorithm where the agent with the lowest rewards for hotSpot visitations (and the previously mentioned factors as well) gets priority for visiting the most "needy" hotSpot (the one with least visits).

1.3 Formulation

The challenge discussed in this paper is to allow multiple fully autonomous agents to move around in an unknown terrain in order to explore and optimize their *objective function* F. The overall *objective* is to coordinate paths amongst the agents in order to re-visit the discovered hotSpots capturing the nonlinear constraints of: 1) the intensity for every hotSpot with respect to other hotSpots; 2) the energy remaining in the agent(s); and 3) the exploration-exploitation variable factor that determines the size of the planning cycle's step size which, in turn, defines the exploratory pattern sizes.

System constraints

- Agent's sensing radius is restricted to 2 space units around it.
- Different terrain textures and elevations incur different mobility costs.

- hotSpots should be visited proportionally to their average intensity value.
- SAs must balance between their exploration and exploitation ratios.
- Q_t = summation of the all the system requirements and constraints.

Decision Vvariables

- Let I_K represent the value of the *intensity* of hotspot K.
- Let V_K represent the (total) number of *visits* to hotspot K.
- Let V_{Ki} represent the number of *visits* to hotspot K by agent i, so $V_K = \sum_i V_{Ki}$.
- Let $V = \sum_K V_K$ represent the total visits to all hotspots.
- We obtain $\overline{V}_K = \frac{V_K}{V}$ by normalizing V_K. Similarly,
- We obtain $\overline{I}_K = \frac{I_K}{I}$ where $I = \sum_K I_K$.

To satisfy the constraints and conditions set of the mission, the objective function should be maximized by visiting the highest intensity hotspot, in other words it should have the form of $\overline{V}_K \propto \overline{I}_K^q \Rightarrow \log(\overline{V}_K) \propto q \cdot \log(\overline{I}_K)$, which is satisfied for q to be 1, therefore $\frac{\overline{V}_K}{\overline{I}_K} = 1$. This in turn defines the range of values for the optimization function J from $[(-1) \rightarrow (+1)]$, where $J = 1$ is value for the most optimal and $J = (-1)$ for the least. This also stipulates that the total values of the K hotspots $\overline{I}_K$ should sum up to 1.

The optimization can be rewritten as: $J = \frac{1}{N_K} \cdot \sum_K (1 - |1 - \frac{\overline{V}_K}{\overline{I}_K}|)$ where N_K represents the number of hotspots K.

1.4 Simulation Results and Conclusion

We conducted many simulation runs of our system on a $(100 \times 100 \times 5)$ unit 3D terrain. We tested many types of terrain elevations and textures and varied the number of hotSpots and SAs. We use 3 types of SAs that we gave identical physical values (energy, sensing radius, ...) . The first type of SAs where ones that used a completely random (and selfish) algorithm, the second was semi-aware (with limited communication radius) and the third was the fully aware kind. Several conclusions can be drawn from the results we collected. First on the coverage (exploration) issue, the third type outperformed all the other types, which was expected, and the results can be seen in graph 1.1. The random SAs outperformed the other 2 types at the beginning, but soon flattened out. On the other hand we see a repetitive "pattern" in the semi-aware case, while we

only see a similar pattern, half-way through simulations for the fully aware case. Finally we see that the fully aware agents were able to discover about 50*percent* of the terrain. In the next graph 1.2, we study one facet of exploitation which is

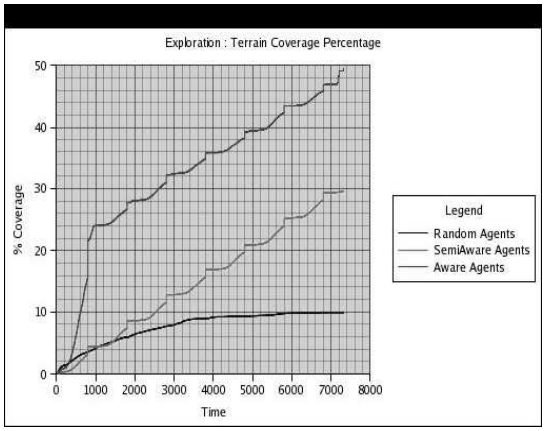

Figure 1.1: Terrain coverage

the total number of visits all hotSpots received. The results show that the semi-aware SAa outperformed the random and aware SAs. But this does not mean that the constraints of the system on the hotSpots and on the SAs were optimally met, as will be revealed in the following 2 graphs 1.3 and 1.4. Although it does show a great improvement over the random case, the localized agents have a tendency of "over-doing" the exploitation as exploitation of the "same" hotSpots occurs frequently due to the restrictions of communication. In graph 1.3 we see a big discrepancy between the most visited hotSpot and the least visited one for the semi-aware agents, something that is less acute for the fully aware SAs.

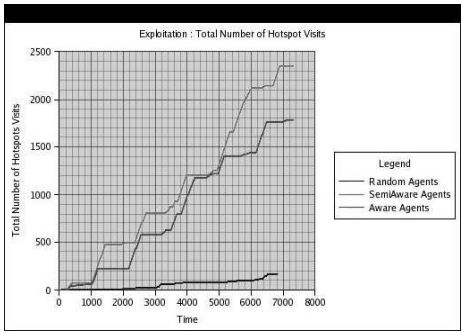

Figure 1.2: Total number of hotSpot visitation for 5 agents and 20 hotSpots

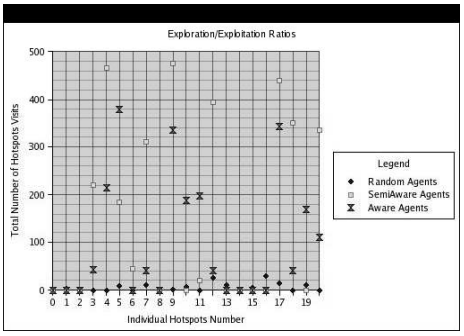

Figure 1.3: Individual hotSpots visitation number for 5 agents and 20 hotSpots

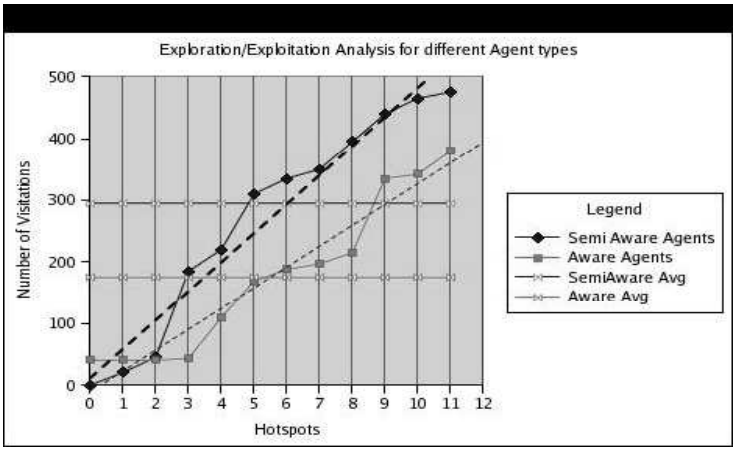

Figure 1.4: Analysis with average and linear regression for the semi-aware and aware SAs results depicted in graph 1.3

This fact shows that the exploration/exploitation ratio was not as balanced in the semi-aware case as it was in the fully aware case. This is shown once again in graph 1.4 where only the semi-aware and aware results from graph 1.3 are ordered and analyzed.

Bibliography

[1] Bianchi, Leonora, "Notes on dynamic vehicle routing - the state of art", *Tech. Rep. no. 1*, IDSIA Switzerland, (2000).

[2] JPL, "Jupiter's europa harbors possible warm ice or liquid water", *Tech. Rep. no. 1*, JPL, (2001).

[3] JPL, "Mars exploration rover mission", *Tech. Rep. no. 1*, JPL, (2003).

[4] NASA, "Listening for an ocean on europa", *Tech. Rep. no. 1*, NASA, (2001).

[5] Simonetti, Neil, "Applications of a dynamic programming approach to the traveling salesman problem", *Tech. Rep. no. 1*, Carnegie Mellon, (1998).